

Charan Surya Nikhil Kilana

469-734-3401 | charankilana2025@gmail.com | [GitHub](#) | [Portfolio](#)

Professional Summary

DevOps Engineer with 5 years of experience in automating infrastructure, CI/CD pipelines, and cloud-native deployments across Azure, AWS and Kubernetes environments. Proficient in Terraform, Ansible, Jenkins, GitHub Actions, ArgoCD, Docker and Kubernetes, with expertise in building scalable, secure, and cost-optimized infrastructure, as well as migration projects and observability, using Prometheus, Grafana, CloudWatch, Azure Monitor.

Education

GITAM University India

Bachelor of Technology, Computer Science

May 2017 - June 2021

Skills

Languages, Scripting & Frameworks: Java, Python, Bash, PowerShell, Groovy, YAML, XML, JSON

Cloud Services: Azure (Azure DevOps, VMs, Blob, ACR, ACS, AKS, VNets, Key Vaults, Azure Monitor) and AWS (EC2, EBS, S3, ECS, ECR, EKS, Route53, CloudWatch, Lambda, SQS, SNS)

DevOps Tools: Linux (RHEL, Amazon Linux), Git (GitHub, GitLab, Bitbucket), Jenkins, Azure DevOps, GitHub Actions, Maven, Nexus, Docker, Kubernetes, ArgoCD, Ansible, Terraform

Observability & Methodology: Prometheus, Grafana, EFK (Elasticsearch, Fluentd, Kibana), Splunk, Slack
Agile/Scrum

Certification

[HashiCorp Certified Terraform Associate](#), [AWS Certified Solution Architect - Associate](#), [AWS Certified Cloud Practitioner](#)

Professional Experience

Recreational Equipment, Inc (REI), Dallas, TX

August 2023 – present

DevOps Engineer

- Assisted in migrating **80+ microservices** to **Amazon EKS**, enabling scalable infrastructure and secure microservice communication via **Istio service mesh** and integrated **RabbitMQ/SQS** to increase throughput by 2x.
- Implemented **ArgoCD** with Custom Resource Definitions (CRDs) on **Amazon EKS** to automate Helm-based deployments, reducing manual effort by 70% and ensuring consistent, reliable rollouts.
- Implemented robust observability for **Kubernetes** clusters with **Prometheus & Grafana** and **Amazon CloudWatch** for SLA/SLO based metrics and alerting, along with the **EFK** stack for centralized logging and RBAC controls, achieving a 40% reduction in MTTR.
- Optimized **Jenkins** pipelines, blocking high-impact CVEs from deployment to **Amazon ECR**, reducing release delays by 30% while ensuring production security compliance.
- Automated infrastructure provisioning using **Terraform**, ensuring consistent, repeatable, and scalable deployments across multiple environments.
- Managed cloud with **Terraform** for **EKS** clusters, preventing state conflicts via remote state management with **Amazon S3** and **DynamoDB** state locking, enhancing infrastructure reliability.
- Implemented monitoring solutions using **Prometheus** and **Grafana**, tracking system performance, collecting metrics, and visualizing data for effective decision-making.
- Automated repetitive tasks and manual interventions using scripting languages and configuration management tools using **Ansible**, enhancing operational efficiency and reducing human error.
- Implemented **AWS Lambda** functions in **Python** triggered by **CloudWatch** Events to enforce **S3/EBS** security, auto-heal **EC2/EKS** workloads, and automate cost optimization, reducing cloud spend by 20%.
- Leveraged **CAST AI** on **Kubernetes** for intelligent downscaling and autoscaling, achieving up to 45% cost reduction within two months.

Azure DevOps Engineer

- Collaborated with development and operations teams to streamline release cycles, improve deployment processes, and ensure smooth operation of the software development lifecycle.
- Automated provisioning of **160+ Azure Linux (RHEL) VMs** using **Terraform** and configured WebSphere applications with **Ansible** playbooks, reducing deployment time from days to a few hours and ensuring environment consistency across subscriptions.
- Implemented **Azure DevOps** pipelines integrating **SonarQube** (SAST), DAST, and **Trivy** security scans, with EAR artifacts versioned and stored in **ACR**, reducing release delays by 40% and ensuring secure, production-ready builds.
- Containerized **Java** and **Python** microservices using multi-stage Docker builds integrated into **Azure DevOps CI/CD**, reducing image size by 70% and improving deployment speed and efficiency.
- Designed reusable **Terraform** modules for provisioning **AKS** clusters, **VNets**, **Key Vaults**, and other Azure resources, reducing provisioning errors by 60% and enabling consistent infrastructure deployments.
- Configured **Azure Application Gateway** with Ingress Controller on **AKS** to serve as a reverse proxy and load balancer, improving response times by 30% and enhancing application availability.
- Configured **Kubernetes** resources including **Pods**, **Deployments**, **Services**, and **Ingress**, **PV & PVC** implementing best practices for container orchestration.
- Troubleshoot and resolve Kubernetes-related issues, including pod and node-level problems, ensuring minimal downtime and maintaining application reliability.
- Deployed stateful **RabbitMQ** workloads on **AKS** using **Azure Managed Disks** for Persistent Volumes, ensuring data durability and preventing message loss during pod restarts or node failures.
- Optimized **Prometheus** and **Grafana** monitoring by integrating with **Azure Monitor**, reducing metric retention and offloading long-term data to **Thanos**, cutting storage usage by 40% and improving dashboard performance for SLA/SLO tracking.
- Implemented **Blue-Green deployment** strategy for critical services using **Azure DevOps** release pipelines and traffic routing via **Application Gateway**, achieving zero downtime during production rollouts.

Projects

- **End-to-End Multi-Language Microservice Deployment ([LINK](#)) ([LINK](#)):** Automated CI/CD for eight services using **GitHub Actions**, **Jenkins**, and **Docker** with **Slack** notifications. Deployed on **Amazon EKS** via **Helm** and **ArgoCD** with **NGINX Ingress + ALB**. Enhanced observability by integrating **Prometheus-Grafana**, **EFK**, and **Open Telemetry** with **Jaeger**, improving debugging efficiency.
- **Claude AI Model Context Protocol Integration ([LINK](#)):** Built and integrated custom **MCP** servers with **Claude AI** to enable real-time external data retrieval, streamlining AI-augmented workflows and enhancing automation.
- **ML Pipeline with DVC & MLflow ([LINK](#)):** Built **ML** workflows using **DVC** for dataset/model versioning with **S3** as remote storage. Integrated **MLflow** for experiment tracking, logging metrics, and managing model artifacts, deployment through **GitHub** Actions pipelines.